A New Look at the Conditions for the Synthesis of Speed-independent Circuits

Enric Pastor, Jordi Cortadella and Oriol Roig
Department of Computer Architecture
Universitat Politècnica de Catalunya
08071 Barcelona, (Spain).

Abstract

This paper presents a set of sufficient conditions for the gate-level synthesis of speed-independent circuits when constrained to a given class of gate library. Existing synthesis methodologies are restricted to architectures that use simple AND-gates, and do not exploit the advantages offered by the existence of complex gates. The use of complex gates increases the speed and reduce the area of the circuits. These improvements are achieved because of: 1) the elimination of the distributivity, signal persistency and unique minimal state requirements imposed by other techniques; 2) the reduction in the number of internal signals necessary to guarantee the synthesis; and finally 3) the utilization of optimization techniques to reduce the fan-in of the involved gates and the number of required memory elements.

1 Introduction

All the potential benefits of asynchronous circuits may be useless because of the presence of *hazards* (transient errors due to stray delays). Most of the existing techniques rely on a high-level specification of the system, and derive logic-level or gate-level networks that are *hazard-free* under the assumption of some delay model.

Burst-mode design methodologies [8, 16] use transformations [6, 9] or technology mapping [13] to create a specific gate-level network. *Speed-independent* design methodologies from event models [3, 14] assume the existence of an extensive library of complex gates without internal hazards to avoid the technology mapping step. Varshavsky *et al.* [14] derived *speed-independent* circuits from *distributive* State Transition Diagrams using basic gates, but producing large and inefficient circuits.

Beerel *et al.* [2] extended Varshavsky's methodology to a *Sum-of-Products* (SOP) architecture. This method allows input choices and applies some optimizations, but does not use a set of necessary conditions to guarantee a correct solution. Kondratyev *et al.* [5] have defined sufficient conditions to decide when a speed-independent SOP implementation can be derived. Both techniques impose two conditions that restrict the number of specifications that can be implemented and the optimizations that can be applied: 1) the use of SOP architectures that can only include AND-gates, OR-gates, and C-elements (where *distributivity* and *signal persistency* are necessary conditions); and 2) the *unique entry condition* that does not allow the use of certain types of input choices. Finally, Siegel *et al.* [12] have developed a technique to decompose high fan-in AND-gates into a multi-level structure.

This work presents a set of sufficient conditions for the gate-level synthesis of speed-independent circuits when restricted to a given gate-library. The proposed technique takes advantage of using complex gates, such as AOI-gates and OAI-gates, typically available in CMOS libraries [1]. The *distributivity*, *signal persistency* and *unique entry* conditions can be safely eliminated as restrictions, increasing the specifications that can be successfully implemented. Transformation techniques to reduce the number of *non-persistent* signal transitions, and the number of *minimal* (or *maximal*) states are introduced. These transformations together with the *Unique State Coding* property, control the type and complexity of the gates involved in the implementation. The proposed synthesis conditions reduce the delay, area and memory elements in speed-independent realizations. Several clarifying examples of circuit specifications and speed-independent realizations, even not satisfying the previously required conditions, are presented.

2 Definitions

A *State Transition Diagram* (STD) [14, 15] is a quadruple $\langle S, E, A, \lambda \rangle$, where S is a set of states, $E \subseteq S \times S$ is a set of transitions, A is a set of signals, and $\lambda : S \rightarrow \mathrm{B}^{|A|}$ is a total labeling function that encodes each state with a *binary vector* of signal values. The set of signals $A = A_I \cup A_O$ is divided into input signals A_I, and non-input signals A_O. Rising (falling) transitions of signal a_i are denoted a_i+ (a_i-), and generic transitions a_i*. An STD (*Example 1*) is described in Fig. 2(a), where $S = \{s_1, \ldots, s_{21}\}$, $E = \{(s_1, s_2)(s_2, s_3)(s_3, s_4)\ldots\}$, $A = \{a, b, c, d, e, f\}$, and $\lambda(s_1) = (000000)$, $\lambda(s_2) = (100000)$, and so on.

For every arc connecting a pair of states s and s', their label differs exactly in one signal value, say the ith. Then, a_i is called *excited* in the state s. Signals not *excited* in a state are called *stable*. For any pair $s, s' \in S$, s' is *reachable* from s if there is a sequence $a_1*, \ldots, a_n*$ leading from s to s', denoted by $sE(a_1*, \ldots, a_n*)s'$. $\Delta : 2^S \longrightarrow 2^S$ is the *one-step transition function* that given a set of states X, its image is the set of states $X' = \Delta(X)$ such that for any pair $s \in X$ and $s' \in X'$ then sEs'. No state can have two outgoing transitions labeled with the same signal but with different signs. Moreover, no state s can have an outgoing transition labeled with a rising (falling) transition of a signal a_i, being the encoding function $\lambda(s)_i = 1$ ($\lambda(s)_i = 0$). Such an STD is called *consistently encoded*.

The behavior of a circuit can be represented by an STD. The reachable states depend on the delay model assumed for the gates and wires in the circuit. Two delay models specify the amount of "memory" of the delay (*inertial* and *pure*), while two other define the time required to propagate changes through the delay (*unbounded* and *bounded*). Speed-independent synthesis assumes a negligible delay in the wires, and also the *pure delay* and *unbounded delay* models in the gates [14, 5]. An STD is said to be *speed-independent* under the *pure* and the *unbounded* delay models iff it is output *semi-modular* [15]. Henceforth we will assume that all the STD specifications are *semi-modular* and *consistently encoded*.

The *rising (falling) excitation set* of a_i is the set of states in which some transition a_i+ (a_i-) is *excited* (denoted by $\mathrm{ES}^+(a_i)$ and $\mathrm{ES}^-(a_i)$). The *one (zero) quiescent set* of a_i is the set of states in which a_i has the value 1 (0) and no transition a_i* is

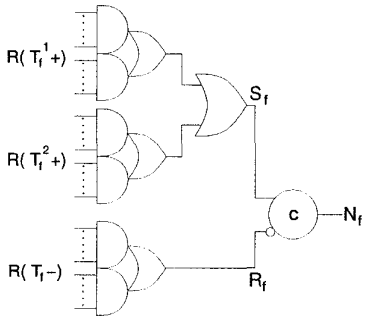

Figure 1: Signal network in a SOF architecture for signal f.

excited (denoted by $\mathrm{QS}^1(a_i)$ and $\mathrm{QS}^0(a_i)$). The *excitation region* $\mathrm{ER}(a_i*)$ is the maximal connected set of states in which the transition a_i* is *excited*. The *quiescent region* $\mathrm{QR}(a_i*)$ is the maximal connected set of states that can be reached from $\mathrm{ER}(a_i*)$, and the *backward quiescent region* $\mathrm{BR}(a_i*)$ is the maximal connected set of states that can reach $\mathrm{ER}(a_i*)$, in both cases without enabling any other transition $a_i'*$.

A transition a_i* is a *predecessor* of $a_i'*$ if there exists an allowed sequence $\{a_i*, \sigma, a_i'*\}$ in which no other $a_i''* \in \sigma$. Conversely, $a_i'*$ is a *successor* of a_i*. The set of predecessors (successors) of a_i* is denoted by ${}^\bullet a_i*$ ($a_i*{}^\bullet$). A state s is *minimal (maximal)* for $\mathrm{ER}(a_i*)$ if s has no predecessors (successors) within the region. An STD satisfies the *unique entry condition* iff all the excitation regions have exactly one minimal state. A transition a_j* (and the signal a_j) is *trigger* for a_i* if $\mathrm{ER}(a_i*)$ can be entered by executing a_j*. A signal a_j is *ordered* with a_i* if no a_j* is excited in $\mathrm{ER}(a_i*)$, otherwise it is *concurrent*. A *trigger* a_j* is *signal non-persistent* with a_i* if a_j* is *concurrent* with a_i*, otherwise it is *signal persistent*.

An STD satisfies the Complete State Coding (CSC) property iff, when the same binary code is assigned to two different states, the transitions of the non-input signals enabled at both states are identical, that is $\forall a_i \in A_O$: $[\mathrm{ES}^+(a_i) \cap \mathrm{QS}^0(a_i) = \emptyset] \wedge [\mathrm{ES}^-(a_i) \cap \mathrm{QS}^1(a_i) = \emptyset]$. If all the states of the STD are assigned a unique binary code the *Unique State Coding* (USC) property holds. Finally, the set of vertices in $\mathrm{B}^{|A|}$ that do not correspond to any state in S is called the *don't care-set* (*dc-set*).

3 Implementation Architecture Overview

The implementation strategy assumed in this paper is named a two-level *Sum-of-Functions* (SOF) architecture. Following this architecture, a *signal network* (N_{a_i}) is created for every output signal. The transitions of a_i are grouped into *transition clusters* that only contain rising (falling) transitions, denoted T_i+ and T_i-. N_{a_i} consists of a first level of rising and falling *region covers*, $R(T_i+)$ ($R(T_i-)$). The *region cover* is a function that covers the states in the excitation regions and may cover some of the states in the *quiescent regions* and *backward quiescent regions*. The *region covers* are implemented by one AND-gate in a single cube cover, or by complex gates in a multiple cube (*poly-term*) cover. The rising *region covers* are combined with an OR-gate to create the *set region network* (S_{a_i}), and the falling *region covers* create the *reset region network* (R_{a_i}). Finally, both region networks implement N_{a_i} using a Muller's C-element.

Fig. 1 depicts a signal network for f. S_f is composed of region covers for $T_f^1+ = \{f+/1\}$ and $T_f^2+ = \{f+/2\}$, while R_f is composed of a region cover for $T_f- = \{f-\}$. All region covers are assumed to be implemented with AO-gates.

4 Speed-independence Conditions

This section presents a sufficient condition, called *monotony*, for the synthesis of speed-independent circuits following the SOF architecture model. We refer to [11] for a deeper analysis.

Definition 1 *We define the* **transition clusters** *of one output signal a_i to be a total partition $\{T_i^1+, \ldots, T_i^m+, T_i^1-, \ldots, T_i^m-\}$, of its transitions, such that every rising (falling) transition must be in one transition cluster strictly composed of rising (falling) transitions.*

A generic transition cluster is denoted by T_i*. Similarly to transitions, we define the *predecessor* and *successor transition clusters*. The set of predecessor (successor) transition clusters of T_i* is denoted ${}^\bullet T_i*$ ($T_i*{}^\bullet$). In the *Example 1* both T_f^1+ and T_f^2+ are predecessor transition clusters of T_f-, while T_f- is the only predecessor transition cluster for T_f^1+ and T_f^2+.

The regions of a transition are also extended to transition clusters. The *excitation region* $\mathrm{ER}(T_i*)$ is the union of the excitation regions for all the transitions in the cluster (similar extensions exist for $\mathrm{QR}(T_i*)$ and $\mathrm{BR}(T_i*)$). The excitation regions for the transition clusters of the output signal f are depicted in Fig. 2(a). The quiescent regions can be directly derived from them:

$$\begin{aligned} \mathrm{QR}(T_f^1+) &= \{s_6, s_7, s_8, s_9, s_{10}, s_{20}\}, \\ \mathrm{QR}(T_f^2+) &= \{s_{15}, s_{16}, s_{17}, s_{18}, s_{19}, s_{20}\}, \\ \mathrm{QR}(T_f-) &= \{s_1, s_2, s_3, s_4, s_{11}, s_{12}, s_{13}\}. \end{aligned}$$

Definition 2 *A set of cubes is said to be the* **region cover** *$R(T_i*)$, iff for every $s \in \mathrm{ER}(T_i*)$, there exists a cube $c \in R(T_i*)$ that covers $\lambda(s)$. A region cover can also be interpreted as a set of states such that $s \in R(T_j*)$ if its binary code $\lambda(s)$ is covered by any of the cubes.*

Region covers can be derived for the transition clusters in *Example 1*: $R(T_f^1+) = \{101110\}$, $R(T_f^2+) = \{011110\}$, and $R(T_f-) = \{000000\}$.

A *region cover* $R(T_i*)$ may also cover successor states in $\mathrm{QR}(T_i*)$, or predecessor states in $\mathrm{BR}(T_i*)$.

Definition 3 *$R(T_i*)$ is said to be* **feasible** *iff does not cover any state outside $\mathrm{MR}(T_i*) = \mathrm{BR}(T_i*) \cup \mathrm{ER}(T_i*) \cup \mathrm{QR}(T_i*)$.*

However, it is possible to have a state included in the *quiescent regions* or the *backward quiescent regions* for several transition clusters. In the *Example 1*, the state s_{20} is shared by both $\mathrm{QR}(T_f^1+)$ and $\mathrm{QR}(T_f^2+)$ and therefore cannot be covered by any of the region covers $R(T_f^1+)$ and $R(T_f^2+)$. If any of both region covers, $R(T_f^1+)$ or $R(T_f^2+)$, covers s_{20} the resulting signal network cannot be speed-independent. In general, any state can be covered by only one of the region covers for the rising (or falling) transition clusters of an output signal. States that are included in the quiescent regions (or the backward quiescent regions) of more than one transition cluster cannot by covered. Therefore *feasible* region covers that do not include states shared by the quiescent regions (or backward quiescent regions) of other transition clusters are called *one-hot encoded*. *One-hot encoded* transition clusters guarantee that, at any time, only one of the regions evaluates to 1.

Definition 4 *The* **restricted quiescent region** *$\mathrm{QR}^r(T_i*)$ is the subset of $\mathrm{QR}(T_i*)$ such that:*

$$\mathrm{QR}^r(T_i*) = \mathrm{QR}(T_i*) - \bigcup_{a_i* \notin T_i*} \mathrm{QR}(a_i*).$$

Definition 5 *The* **restricted backward quiescent region** *$\mathrm{BR}^r(T_i*)$ is the subset of $\mathrm{BR}(T_i*)$ such that:*

$$\mathrm{BR}^r(T_i*) = \mathrm{BR}(T_i*) - \bigcup_{a_i* \notin T_i*} \mathrm{BR}(a_i*).$$

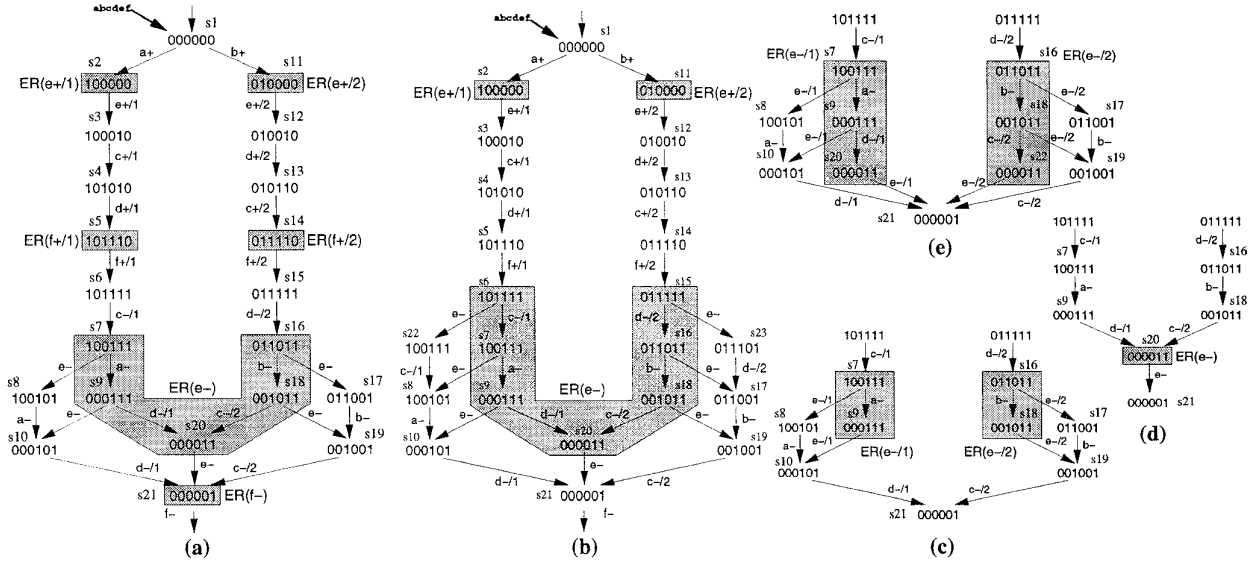

Figure 2: *Example 1*: (a) non-persistent, (b) persistent STD; (c) persistency, and (d) concurrency constraints, (e) region splitting.

Definition 6 *The feasible region cover $R(T_i*)$ is said to be* **one-hot encoded** *iff $R(T_i*)$ does not cover any state outside* $\mathrm{MR}^r(T_i*) = \mathrm{BR}^r(T_i+) \cup \mathrm{ER}(T_i+) \cup \mathrm{QR}^r(T_i+)$.

Covering states in the backward quiescent regions is only possible because of the special characteristics of the C-element. If the output of the C-element ($a_i = S_{a_i}\overline{R_{a_i}} + a_i(S_{a_i} + \overline{R_{a_i}})$) is stable at 0 and R_{a_i} is still at 1, the value of S_{a_i} does not influence the behavior of the output. Therefore S_{a_i} can be safely switched to 1 before the excitation region $\mathrm{ER}(T_i+)$. Similar conditions can be stated for the falling transition clusters T_i-.

A *one-hot encoded* $R(T_i*)$ is *correct* if any state in $\mathrm{BR}(T_i*)$ covered by $R(T_i*)$ it is also covered by the region cover of a predecessor transition cluster of T_i*. Any state in $\mathrm{BR}(T_i*)$ and also contained in the quiescent regions of several transition clusters in ${}^\bullet T_i*$ cannot be covered by any of its region covers, and therefore this state cannot be covered by $R(T_i*)$ neither.

Definition 7 *The one-hot encoded $R(T_i*)$ is* **correct** *iff* $\forall s \in R(T_i*)$: $s \in \mathrm{BR}(T_i*) \Leftrightarrow \exists! T_j' * \in {}^\bullet T_i* : s \in R(T_j'*)$.

Theorem 8 [11] *A correct region cover can be derived for each transition cluster T_i* iff the STD has the CSC property.*

Definition 9 *The correct region cover $R(T_i*)$ is said to be* **monotonic** *iff $R(T_i*)$ changes exactly twice in any sequence, where:*

1. *the rising change is at a state in* $\mathrm{BR}^r(T_i*) \cup \mathrm{ER}(T_i*)$,
2. *the falling change is at a state in* $\mathrm{QR}^r(T_i*)$.

Lemma 10 [11] *A monotonic rising transition of $R(T_i*)$ is guaranteed if given any state s in $\mathrm{BR}^r(T_i*)$ covered by $R(T_i*)$:*

$$\forall s' \in S \ \wedge \ sEs' \ : \ s \in \mathrm{BR}^r(T_i*) \ \Leftrightarrow \ s' \in R(T_i*).$$

Similarly, a monotonic falling transition of $R(T_i*)$ is guaranteed if given any state s in $\mathrm{QR}^r(T_i*)$ covered by $R(T_i*)$:

$$\forall s' \in S \ \wedge \ s'Es \ : \ s \in \mathrm{QR}^r(T_i*) \ \Leftrightarrow \ s' \in R(T_i*).$$

Note that the SOF architecture requires that no cube in $R(T_i*)$ can contain the literal corresponding to a_i. Otherwise a feedback is introduced into the signal network.

Proposition 11 [11] *It is possible to derive a correct region cover $R(T_i*)$ such that none of its cubes contains the literal corresponding to a_i only if*

$$\forall s \in \mathrm{ER}(T_i*) \ : \ sE(a_i*)s' \wedge a_i* \in T_i* \wedge s' \in \mathrm{QR}^r(T_i*).$$

According to Prop. 11, no cube in the region covers for the output signal f (see Fig. 2(a)) can contain the literal corresponding to f, and therefore the region covers $R(T_f^1+) = \{10111\text{-}\}$, $R(T_f^2+) = \{01111\text{-}\}$, and $R(T_f-) = \{00000\text{-}\}$ are *monotonic*.

Definition 12 *A correct region cover $R(T_i*)$ is said to be a* **complete region cover** *iff all (but only) the states in* $\mathrm{ER}(T_i*) \cup \mathrm{QR}(T_i*)$ *are covered by at least one cube in $R(T_i*)$.*

A signal network that only uses rising (or falling) transition clusters, called a *complete signal network*, can be derived if all the rising (or falling) region covers are complete. When all the rising region covers are complete, the set region network is implemented while the reset region network and the C-element are removed (similarly for the falling region covers).

Theorem 13 [11] *A SOF circuit implementation derived from monotonic region covers $R(T_i*)$ is speed-independent.*

The number of clusters determines the number of gates used to implement the signal networks. Two clusters can be used, one containing the rising transitions and the other containing the falling transitions. Then the signal network is reduced to two complex gates for the set and reset region networks, and the C-element. If every cluster contains exactly one transition then, the signal network contains one gate for each transition, two OR-gates and the C-element.

Lemma 14 [11] *The correct region covers $R(T_i*)$ of an output signal a_i are monotonic only if:*

$$\begin{aligned} \forall\, a_i^k* \in T_i^k*,\ a_i^l* \in T_i^l* : \exists s, s' \ : \\ s \in \mathrm{MR}(a_i^k*) \ \wedge \ s' \in \mathrm{MR}(a_i^l*) \ \wedge \\ \lambda(s) = \lambda(s') \ \wedge \ \lambda(s) \sqsubseteq R(T_i^k*) \implies T_i^k* = T_i^l*. \end{aligned}$$

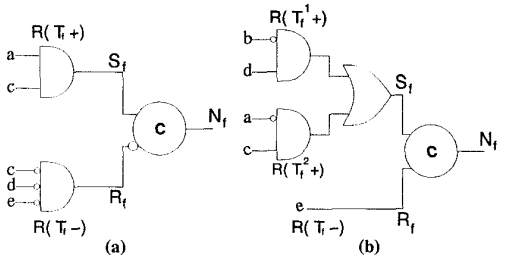

Figure 3: Signal networks for signal f in the *Example 1*.

The state encoding highly influences the structure of the clusters. Assuming that a_i^k+ is excited at s, that a_i^l+ is excited at s' and that $\lambda(s) = \lambda(s')$. $\mathrm{ER}(a_i^k+)$ and $\mathrm{ER}(a_i^l+)$ will intersect, and both transitions must be in the same transition cluster to guarantee *monotony*.

Lemma 15 [11] *Given the transition clusters in such a way that* $|T_i*| = 1$, *then there exist correct region covers* $R(T_i*)$ *that are monotonic only if the STD satisfies the USC condition.*

5 Region Cover Minimization

A region cover can be simplified by reducing the number of cubes in it, and the literals in each cube, increasing the probability of finding a match in a gate-library. In any case, after a minimization the region cover must remain monotonic.

5.1 Monotony Verification

The monotony verification technique consists of three steps:

1. *One-hot encoding verification*, to guarantee that only those states available for minimization have been used.
2. *Correctness verification*, to guarantee that the region cover correctly overlaps with the predecessor region covers.
3. *Monotonic transition verification*, to guarantee that the region cover changes exactly twice.

Proposition 16 [11] $R(T_i*)$ *is not monotonic iff any of the following requirements is not satisfied:*

1. $R(T_i*) \cap \left(S - \mathrm{MR}^r(T_i*)\right) = \emptyset$.
2. $R(T_i*) \cap \mathrm{BR}(T_i*) \subset \bigcup_{\forall T_i'* \in {}^\bullet T_i*} R(T_i'*)$.
3. $\Delta^{-1}\left(R(T_i*) \cap \mathrm{QR}^r(T_i*)\right) \subset R(T_i*)$;
   $\Delta\left(R(T_i*) \cap \mathrm{BR}^r(T_i*)\right) \subset R(T_i*)$.

5.2 Signal Network Synthesis and Minimization

Four different techniques can by applied to simplify a region cover. After any simplification, the region covers can be checked to eliminate all the redundant cubes.

Forward region expansion: The cover is expanded towards the restricted quiescent region, increasing the states contained in it. The final objective is to derive a *complete* region cover.

Backward region expansion: The cover is expanded towards the restricted backward quiescent region. All the newly covered states must be covered by the region covers of the predecessor transition clusters.

DC region expansion: A cover is expanded to the *dc-set* of the specification when a literal is removed and no new states are covered.

Region merging: Transition clusters can be merged together when the size of the cubes (or its number) in the resulting region cover decreases.

```
speed-independent_SOF_implementation (STD, gate-library) {
    foreach non-input signal a_i do{
1       foreach transition a_i* do T_i* = {a_i*};
        foreach transition cluster pair T_i^l*, T_i^k* do
            if ( ER(T_i^l*) ∩ ER(T_i^k*) ≠ ∅ ) merge(T_i^l*, T_i^k*);
2       foreach transition cluster T_i* do
            R(T_i*) = compute_irredundant_cover(T_i*);
3       foreach transition cluster T_i* do
            forward_region_expansion(R(T_i*));
4       foreach transition cluster T_i* do
            backward_region_expansion(R(T_i*));
5       foreach transition cluster T_i* do
            dc_region_expansion(R(T_i*));
6       N_a_i = create_signal_network();
7       foreach transition cluster T_i* in N_a_i do
            map_region_cover(N_a_i, R(T_i*), gate-library);
}   }
```

Figure 4: Speed-independent signal networks synthesis algorithm.

Two correct implementations can be derived for signal f depending on the order in which the minimizations are applied.

Signal network A (see Fig. 3(a)):

$R(T_f^1+)$ and $R(T_f^2+)$ can be expanded to the *dc-set* by removing $\{a\}$ ($R(T_f^1+)$ = {-0111-} and $R(T_f^1+)$ = {-1111-}). Both covers are merged. The result is a single cluster T_f+ with $R(T_f+)$ = {--111-} in which $\{b\}$ has been removed. Finally, $R(T_f+)$ is expanded to the *dc-set* by removing $\{e\}$ ($R(T_f+)$ = {--11--}). $R(T_f-)$ is expanded to the quiescent region by removing $\{a, b\}$ ($R(T_f-)$ = {--000-}).

Signal network B (see Fig. 3(b)):

A different solution can be obtained initially expanding $R(T_f^1+)$ and $R(T_f^2+)$ to the quiescent region by removing $\{c, a, e\}$ and $\{d, b, e\}$ respectively. The resulting covers are $R(T_f^1+)$ = {-0-1--} and $R(T_f^2+)$ = {0-1---}. $R(T_f-)$ (from network A) is further simplified if it is expanded to the backward quiescent region by removing $\{c, d\}$. The resulting cover is $R(T_f-)$ = {----0-}.

The minimization techniques can be combined into the synthesis algorithm presented in Fig. 4 and divided in seven main steps. The initial transition clusters of cardinality one are computed, and those clusters which excitation regions intersect are merged to preserve the *one-hot encoding* requirement (1). An initial irredundant region cover is computed for each one of the previously defined transition clusters (2). At this point the minimization techniques are applied in the predetermined order (3,4,5). After any transformation all the redundant cubes are eliminated, and transition clusters are checked to be merged if improvements are obtained. Finally, the region covers in the final signal network are created and mapped onto a given gate-library [4] (6,7).

6 Easing the Synthesis Requirements

This section studies four important restrictions imposed by the existing synthesis techniques [2, 5]. The *distributivity* and *signal persistency* requirements are necessary conditions for synthesis if only AND-gates are available [5]. The third, called *unique entry condition*, restricts each excitation region to have a single minimal state. Finally, single cube region covers impose strict state encoding requirements. STD that fulfill the CSC and even the USC conditions may not have an speed-independent implementation.

Definition 17 [5] *The* **cover cube** *for* $\mathrm{ER}(a_i*)$ *is defined to be a cube* $c(a_i*)$ *such that for every literal* $c_j(a_i*)$, *if* a_i *is:*

1. *ordered with* a_i* *then:* $c_j(a_i*) = a_j$ *if* $a_j = 1$ *in* $\mathrm{ER}(a_i*)$, *or* $c_j(a_i*) = \overline{a_j}$ *if* $a_j = 0$ *in* $\mathrm{ER}(a_i*)$,
2. *concurrent with* a_i* *then:* $c_j(a_i*) = *$.

A cover cube $c(a_i*)$ *will be called* **correct** *if it covers all (but only) the states in* $\mathrm{ER}(a_i*)$.

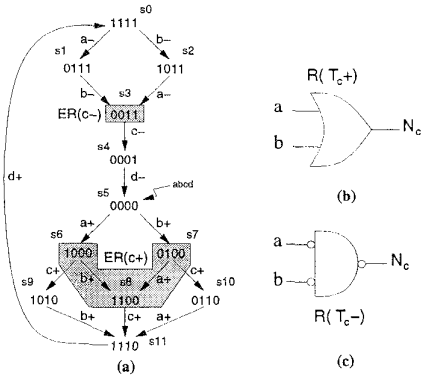

Figure 5: *Example 2*: Non-distributive STD.

6.1 Distributivity

Distributive STDs are a subclass of semi-modular STDs in which the local history for each state is unique. Moreover, *distributivity* is a necessary condition for both the existence of single cube region covers and the *unique entry condition*.

Lemma 18 [14] *In a semi-modular but non distributive STD, there is at least one excitation region with several minimal states.*

Lemma 19 [14] *In a non distributive STD, there is at least one excitation region that cannot be correctly covered by its corresponding cover cube.*

Example 2 (depicted in Fig. 5(a)) presents a non distributive STD. Assuming that c is a non-input signal, then $T_c+ = \{c+\}$ and $T_c- = \{c-\}$. $c(c+) = (\text{--}00)$ may be used to implement $R(T_c+)$. However $c(c+)$ it is not a correct cover cube. The states in $\text{ER}(c+) = \{s_6, s_7, s_8\}$ are covered by $c(c+)$ as well as $s_5 \in \text{BR}(T_c+)$.

The multi cube approach requires two cubes for $R(T_c+) = \{1\text{-}00, \text{-}100\}$ and a single cube for $R(T_c-) = \{0011\}$. Both region covers can be minimized obtaining *complete* signal networks (see Fig. 5(b) and Fig. 5(c)).

6.2 Signal Persistency

The work by Kishinevsky *et al.* [5] proved that signal persistency is a necessary condition to guarantee that a transition cluster of cardinality one can be covered with exactly one cube.

Lemma 20 [5] *Given $T_i* = \{a_i*\}$, the region cover $R(T_i*) = \{c(a_i*)\}$ correctly covers $\text{ER}(a_i*)$ iff a_i* is signal persistent.*

Non-persistent output signal transitions require the transformation of the STD specification by reducing its concurrency. Concurrency reductions may decrease the area of the implementations because they eliminate CSC violations, reducing the number of internal signals, and increase the number of vertices in the *dc-set*. However, they are not desirable because the imposed sequentiality greatly degrades the performance of the system. The use of complex gates allows to derive monotonic region covers for non-persistent transitions.

Example 1 (Fig. 2(a)) depicts a non-persistent transition $e-$. Signals c and d are trigger of $e-$, and transitions $d-/1$ and $c-/2$ are concurrent with $e-$. Therefore, the cover cube $c(e-)$ only contains the literal f, *i.e.* $c(e-) = (\text{-----}1)$. Given $T_e- = \{e-\}$ it can be verified that $R(T_e-) = \{c(e-)\}$ is not monotonic. Both, $s_6 = (101111)$ and $s_{15} = (011111)$ in $\text{ER}(e+/1)$ and $\text{ER}(e+/2)$ are covered by $c(e-)$.

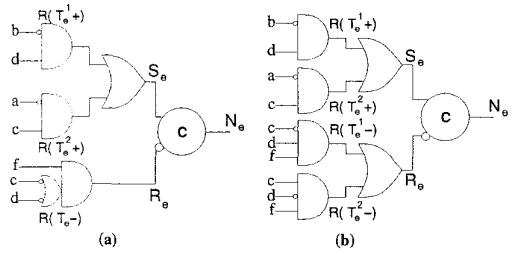

Figure 6: Signal networks for signal e in the *Example 1*.

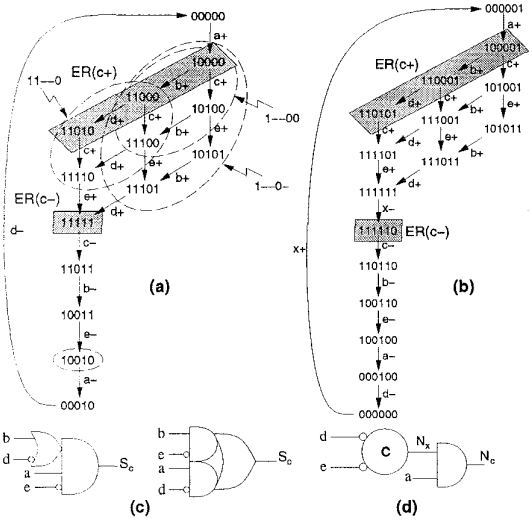

Figure 7: *Example 3*:(a,b) STDs, (c,d) implementations.

The multi cube approach requires four cubes (see Fig. 6(a)) $R(T_e-) = \{1001\text{-}1, 0001\text{-}1, 0110\text{-}1, 0010\text{-}1\}$. The final region cover is $R(T_e-) = \{\text{--}0\text{--}1, \text{---}0\text{-}1\}$, that can be implemented by using an AO-gate with equation $R(T_e-) = (\bar{c} + \bar{d})f$. The rising region covers consist of one cube each ($R(T_e^1+) = \{1000\text{-}0\}$ and $R(T_e^2+) = \{0100\text{-}0\}$). After the minimization both regions are expanded towards the quiescent regions by removing $\{c, d\}$ and to the *dc-set* by removing $\{b\}$ and $\{a\}$ respectively ($R(T_e^1+) = \{1\text{----}0\}$ and $R(T_e^2+) = \{\text{-}1\text{---}0\}$).

6.3 Single Minimal/Maximal State

The notion of *unique entry condition* was introduced in [5] as a necessary condition to derive monotonic cover cubes.

Example 1 in Fig. 2(a) depicts $ER(e-)$ containing two minimal states s_7 and s_{16}. $c(e-) = (\{\text{-----}1\})$ it is not monotonic (as we have seen previously). Increasing the concurrency of the specification by changing the triggers of $e-$ from $\{c-/1, d-/2\}$ to $\{f+/1, f+/2\}$, we obtain the STD depicted in Fig. 2(b). Now the STD contains two more states (s_{22} and s_{23}). However, $ER(e-)$ maintains the same structure with two minimal states s_6 and s_{15}. Now, $c(e-) = (\{\text{-----}1\})$ is monotonic. Therefore the *unique entry condition* it is not necessary for the existence of monotonic cover cubes.

6.4 State Encoding

The monotony conditions when restricted to single cube region covers impose severe constraints in the state encoding of the STD.

A correct cover cube $c(a_i*)$ contains all the states in the excitation region $\mathrm{ER}(a_i*)$, but may also cover other states of the STD. Then, a monotony violation occurs (even if the specification satisfies the CSC and USC requirements), and internal signals must be inserted in the specification.

Example 3 (see Fig. 7(a)) presents an STD that satisfies both CSC and USC. The cover cube $c(c+) = (1\text{---}0)$, covers a state with binary code (10010) in $\mathrm{QR}(c-)$, and therefore it is not monotonic. Given $T_{c+} = \{c+\}$, the complex gate approach will use $R(T_{c+}) = \{11\text{--}0, 1\text{--}00\}$. After the minimization, $R(T_{c+}) = \{\text{-}1\text{--}0, 1\text{--}0\text{-}\}$ is a *complete* region cover that directly implements N_c (see Fig. 7(c)).

6.5 Complete Region Covers

The computation of complete region covers is the most important minimization in the synthesis process because of the C-element elimination. Any synthesis algorithm that only uses single cube region covers will not be able – in most cases – to find complete region covers. Two conditions prevent a quiescent region to be completely covered:

1. When $\mathrm{QR}^r(a_i*)$ does not coincide with $\mathrm{QR}(a_i*)$ (even for a multi cube approach).
2. Any of the successor transitions $a'_i* \in a_i*^\bullet$ has more that one trigger transition.

This second restriction can be eliminated if multi cube region covers are allowed. *Example 1* (see Fig. 2(a)) contains one of these restricted STD specifications. Both $f+/1$ and $f+/2$ have a successor transition $f-$ with a couple of trigger transitions: $\{d-/1, e-\}$ for $f+/1$, and $\{c-/2, e-\}$ for $f+/2$. As an example, $c(f+/1)$ cannot contain literals $\{a, c, d, e, f\}$ to be complete *i.e.* $c(f+/1) = (\text{-}0\text{----})$. It is clear that $c(f+/1)$ covers all the states in $\mathrm{QR}(f+/1)$, but it also covers the state s_{21} of $\mathrm{ER}(f-)$.

This situation is quite close to the *distributivity* restriction presented in Section 6.1. Again, the only possible solution is the use of multi cube region covers. In this particular example a two cube region cover may be used $R(T^1_{f+}) = \{\text{-}0\text{--}1\text{-}, \text{-}0\text{-}1\text{--}\}$.

7 Transformation Techniques

Transformations enforcing *signal persistency*, *unique entry condition*, and to encode the STD are useful to simplify the region covers, increasing the probability of finding a matching library gate.

Non-persistency is eliminated introducing causality relations between an output transition a_i* and its non-persistent concurrent transitions a_j*. Two transformations can be applied enforcing:

1. a_j* to be a successor of a_i*, (*persistency constraint*),
2. a_j* to be a predecessor of a_i*, (*concurrency constraint*).

None of both requires the insertion of state signals, and only the concurrency degree is modified simplifying the structure of the regions. Fig. 2(c) and (d) present the result of applying *persistency* and *concurrency constraints* to $e-$ in *Example 1*.

Transition $e-$ is no longer concurrent with $d-/1$ and $c-/2$ after *persistency constraint*, eliminating the state s_{20}. Two falling transitions $e-/1$ and $e-/2$ are created (see its signal network at Fig. 6(b)). *Concurrency constraint* completely eliminates the concurrency between $e-$ and $a-$, $b-$, $d-/1$ and $c-/2$. States s_8, s_{10}, s_{17} and s_{19} are eliminated, reducing $\mathrm{ER}(e-)$ to s_{20}. The final implementation is composed of $N_e = \mathrm{NOR}(a, b, c, d)$.

Monotony violations can be interpreted as encoding conflicts, and therefore eliminated by using standard encoding techniques based on the insertion of state signals [7, 10]. In *Example 3*, the monotony violation can be interpreted as a CSC conflict between a (dummy) state $s = (10010)$ in $\mathrm{ER}(c+)$, and the real state $s' = (10010)$ in $\mathrm{QR}(c-)$. Fig. 7(b) presents a state encoding that solves the coding conflict. A new signal x is created, and two transitions $\{x+, x-\}$ inserted. The final signal network for c and x are depicted in Fig. 7(d).

A multi-minimal (maximal) state excitation region can be split into several sub-regions by unfolding the specification, *i.e.* by duplicating some of its states. The transition will be unfolded into several new transitions. Each one of the unfolded states introduces a USC violation with any of its "equivalent" states. Such USC violations must be eliminated by using existing state encoding techniques. Fig. 2(e) depicts $\mathrm{ER}(e-)$ in *Example 1* split into two new sub-regions $\mathrm{ER}(e-/1)$ and $\mathrm{ER}(e-/2)$. The state s_{20} has been unfolded into the states s_{20} and s_{22}.

8 Conclusions

A sufficient condition for the synthesis of speed-independent circuits from STDs has been presented. *Monotony* is defined in terms of the *regions* in the specification. Moreover, the structure of every signal network is directly reflected on the *monotony* condition. The complex gate paradigm is efficiently implemented by using multi-cube region covers and a technology mapping final step. Many advantages are offered by this synthesis technique: *distributivity*, *signal persistency*, and the *unique entry condition* are no longer necessary; encoding requirements are eased, coming close to the CSC conditions; and better minimization techniques can be applied, obtaining smaller and faster implementations.

Acknowledgments

This work has been supported by the Ministry of Education of Spain (CICYT) under contract TIC 91-1036, Departament d'Ensenyament de la Generalitat de Catalunya, and ACiD-WG (Esprit 7225). We are also indebted to Bill Lin for many valuable discussions.

References

[1] *Scalable CMOS (SCMOS) Standard Cell Library (dlm V2.2)*. The Institute for Technology Develeopment, March 1990.

[2] Peter A. Beerel and Teresa H. Meng. Automatic gate-level synthesis of speed-independent circuits. In *Proc. ICCAD*, November 1992.

[3] Tam-Anh Chu. *Synthesis of Self-timed VLSI Circuits from Graph-theoretic Specifications*. PhD thesis, MIT, June 1987.

[4] K. Keutzer. DAGON: Technology binding and local optimization by DAG matching. In *Proc. DAC*, pages 341–347, June 1987.

[5] A. Kondratyev, M. Kishinevsky, B. Lin, P. Vanbekbergen, and A. Yakovlev. Basic gate implementation of speed-independent circuits. In *Proc. DAC*, 1994.

[6] David S. Kung. Hazard-non-increasing gate-level optimization algorithms. In *Proc. ICCAD*, pages 631–634, 1992.

[7] Luciano Lavagno, Cho W. Moon, Robert K. Brayton, and A. Sangiovanni-Vincentelli. Solving the state assignment problem for Signal Transition Graphs. In *Proc. DAC*, pages 568–572, June 1992.

[8] Steven M. Nowick and David L. Dill. Synthesis of asynchronous state machines using a local clock. In *Proc. ICCD*, pages 192–197, October 1991.

[9] Steven M. Nowick and David L. Dill. Exact two-level minimization of hazard-free logic with multiple-input changes. In *Proc. ICCAD*, pages 626–630, November 1992.

[10] Enric Pastor and Jordi Cortadella. Polynomial algorithms for the synthesis of hazard-free circuits from signal transition graphs. In *Proc. ICCAD*, pages 250–254, November 1993.

[11] Enric Pastor, Jordi Cortadella, and Oriol Roig. A new look at the conditions for the synthesis of speed-independent circuits. Technical Report RR-94/27, UPC/DAC, October 1994.

[12] P. Siegel and G. De Micheli. Decomposition methods for library binding of speed-independent asynchronous designs. In *Proc. ICCAD*, 1994.

[13] P. Siegel, G. De Micheli, and D. Dill. Automatic technology mapping for generalized fundamental-mode asynchronous designs. In *Proc. DAC*, June 1993.

[14] Victor I. Varshavsky. *Self-Timed Control of Concurrent Processes*. Kluwer Academic Publishers, 1990.

[15] A. Yakovlev and L. Lavagno. A unified Signal Transition Graph model for asynchronous control circuit synthesis. In *Proc. ICCD*, pages 104–111, October 1992.

[16] Kenneth Y. Yun and David L. Dill. Automatic synthesis of 3D asynchronous state machines. In *Proc. ICCAD*, pages 576–580, November 1992.